Electronic Version
Stylesheet Version v1.1.1

Description

[NONVOLATILE MEMORY UNIT WITH PAGE CACHE]

Background of Invention

[0001] Field of Invention

[0002] The invention is related to a controller management for a nonvolatile memory. Particularly, the present invention is related to a controller management for a big block nonvolatile memory.

[0003] Description of Related Art

[0004] Nonvolatile memory chips, which include nonvolatile memory arrays, have various applications for storing digital information. One such application is for storage of large amounts of digital information for use by digital cameras, as replacements for hard disk within personal computer (PCs) and so forth. Nonvolatile memory arrays are comprised of various types of memory cells, such as NOR, NAND and other types of structures known to those

of ordinary skill in the art that have the characteristic of maintaining information stored therein while power is disconnected or disrupted.

[0005] FIG. 1 is a block diagram, schematically illustrating architecture of flash memory card. In FIG. 1, the host end 100 can access data stored in a flash disk 102, in which the flash disk 102 includes a control unit 104 and a memory unit 106. A memory unit may include one or more memory chips. In access operation, the host end 100 usually accesses the data in the memory module 106 via the control unit 104 at the requested address. In addition to communicating with the host, the control unit 104 also takes responsibility of managing the memory unit 106. The flash memory storage device is then configured as a drive by the host. FIG. 2 is a mapping table maintained by the control unit. From the host side, such a drive includes a plurality of logical blocks 108, each of which can be addressed by the host. Namely, the host can access all the logical space including logical block 0, logical block 1, and logical block M-1.

[0006] A flash memory chip generally is divided into a plurality of storage units, like blocks which include one or more sectors. As shown in FIG.2, the physical space of the flash

memory module includes physical block 0, physical block1,..., and physical block N-1. The logical space used by the host is always less than the physical space because some of the physical blocks may be defective or used by the controller for managing the flash memory module. One task of the controller is to create the logical space for host access. Indeed, the host can not directly address the physical space so that the controller must maintain the mapping relations between the logical blocks and the physical blocks. Such a mapping information is always called as a mapping table and can be stored in the specific physical blocks or loaded into the SRAM within the controller. If a host asks for reading a particular logical block, the controller will look up the mapping table for identifying which physical block to be accessed, transfer data from the physical block to itself, and then transfer data from itself to the host.

[0007] FIG. 3A is a drawing, schematically illustrating the conventional mapping architecture. The data block and writing block are formed and managed by the control unit. Each of them includes at least one physical block. In FIG. 3A, the logical block 300 is used by the host to write a data into the data block 302. However, since the overhead

arises from erase-then-program architecture, when the data will be re-written into the data block 302, the data is temporarily written to a writing block 304. The function of the data block is to store original data and the writing block is used as a temporary storage for host current write request. When the writing block 304 is, for example, fully written, then a swap action between the data block 302 and the writing block 304 are necessary. FIG. 3B is a drawing, schematically illustrating how to recycle these blocks. The swap operation generally means that the writing block replaces the data block. However, the replaced data block can be considered as an old block so that it will be erased and then become a spare block. The spare block can be allocated out and become a writing block if the control unit needs such a writing block for the host write request.

[0008] With respect to the data block or the writing block, a sector structure is shown in FIG. 4. In one sector, it usually includes a data area 400, such as a size of 512 byte, and an extra area 402, which may include the information of logical block number, system flag, error correction code (Ecc), and so on. FIG. 5 is a drawing, schematically illustrating the mapping relation between the logical block

300, the data block 302 and the spare block 304. In FIG. 5, the logical block No.0 maps to the data block 302 whose physical Block number is 5, and the spare block 304 is located at physical block No. 200h. The mapping table is divided into the logical area and the physical area. For example, the first row shows that the logical block No. 0 is with respect to the data block No. 5, and the spare block No. 200h can be allocated to become a writing block for any one data block. If host asks for writing sector LBA0 now, then the spare block will be allocated to become a writing block, as shown in FIG 6.Moreover, a sector LBA0 will be written into the first position in the writing block. Now, the field for the first empty sector is filled by 1, which means that the first sector of the empty sectors in the writing block 304 is starting at offset 1 for storing LBA 1.

[0009] FIG. 7 is a drawing, schematically illustrating a data mapping relation after a swap action. Referring to FIG. 6, if the sector LBA0 is to be written again, then a swap action is necessary in the conventional method. Because of the flash characteristic, we can not directly write data into current writing block 304 whose physical block No. is 200h so that a swap operation is needed. The swap oper-

ation we have to do now causes time-consuming and reduces the system performance. All the sectors except LBA 0 in data block must be moved to the current writing block, and then the original data block (physical No.5) will be erased so that the current writing block (physical No. 200) becomes the data block, as in FIG. 7. After swap operation, we still need a writing block for the LBA 0 write operation. We can use the just erased physical block No.5 as the current writing block. Also, we can use the other spare block as the current writing block. Eventually, the LBA 0 data will be written into the current writing block and the mapping table should be updated, as FIG. 7. Here, this kind of situation for writing is called a random write.

[0010] FIG. 8 is a drawing, schematically illustrating the access sequence in the conventional method. After writing to the LBA0, as shown in FIG.6, the host requests to write LBA1. The controller will directly write LBA1 into the next page of 512+16 bytes. Such kind of host side sequential write will not result in a random write in flash memory side.

[0011] FIG. 9 is a drawing, schematically illustrating the block structure of a new-type flash memory having big size blocks. For this type of big size flash memory, usually, one block 500 includes, for example, 64 pages, and each

page has four sectors by a size of 2048+64 bytes. Page is the basic unit to be programmed. The writing sequence is similar to the small size flash memory. FIG.10 is a drawing, schematically illustrating the writing procedure for the big size flash memory. In FIG. 10, the logical block 600 has 64 logical pages, and each logical page has four logical sectors; each logical sector size is 512 bytes for storing user data. Likewise, the data block 602 and the writing (W) block 604 have 64 pages, and each page has four sectors; each sector size is 528 bytes for storing user data and extra data. The arrangement is similar to the previous small size flash memory except block size and page size. When the host requests to write to sector LBA0, then the controller will program entire page0 due to page-based programming operation. Thereby, the original sectors LBA1 - LBA3 will be transferred from the D block 602 into the controller, and then host data LBA0 accompanying with LBA1-LBA3 are together written into page 0 of the W block 304. The mapping table stores the status after programming. The empty pointer indicates offset 1 of the W block 604 is the first blank page.

[0012] When the host requests to write to sector LBA1, then the controller has to program page 0 again, since the sector

LBA1 is a part of the page 0 for the big size block. In this situation, a swap operation occurs for this write operation. In FIG. 11, the swap operation is performed between the data block 5 and the W block 200h, in comparison with FIG. 10.

[0013] As previously discussed, the swap operation will reduce the operation speed. However, the conventional management method between the logical block 600, data block 602, and the W block 604 causes the swap operation rather often for the big size flash memory. If the occurrence of swap operation can be reduced, the operation speed certainly can be improved.

SUMMARY OF INVENTION

[0014] The invention provides a method for managing the access procedure, so as to reduce the occurrence of swap operation.

[0015] The invention provides a method for managing the access procedure by employing a page cache block, so as to reduce the occurrence of swap operation.

[0016] As embodied and broadly described herein, the invention provides a method for managing an access procedure for a big size flash memory, comprising using at least one block as a page cache block. When a host requests to

write a data into a W block, the last page that has at least a portion of the data is written into the page cache block. Each page includes multiple sectors.

[0017] The invention also provides a block structure for a big-size flash memory, comprising a logical block, a data block, a W block, and a page cache block. The page cache block stores the latest page with respect to the W block, in which one page includes multiple sectors.

[0018] It is to be understood that both the foregoing general description and the following detailed description are exemplary, and are intended to provide further explanation of the invention as claimed.

Brief Description of Drawings

[0019] The accompanying drawings are included to provide a further understanding of the invention, and are incorporated in and constitute a part of this specification. The drawings illustrate embodiments of the invention and, together with the description, serve to explain the principles of the invention.

[0020] FIG. 1 is a block diagram, schematically illustrating architecture of flash memory card.

[0021] FIG. 2 is a mapping table.

[0022] FIGs. 3A-3B are a drawings, schematically the conventional mapping architecture and how to recycle.

[0023] FIG. 4 is a drawing, illustrating a sector structure.

[0024] FIG. 5 is a drawing, schematically illustrating the mapping relation between the logical block, the data block and the spare block.

[0025] FIG. 6 is a drawing, schematically illustrating the mapping table associating with the logical block, the data block, and the writing block.

[0026] FIG. 7 is a drawing, schematically illustrating a data mapping relation after a swap action.

[0027] FIG. 8 is a drawing, schematically illustrating the access sequence in the conventional method.

[0028] FIG. 9 is a drawing, schematically illustrating the block structure of a new-type flash memory having big size block.

[0029] FIG. 10 is a drawing, schematically illustrating the block structure and the mapping table in a flash memory with a type of big size block.

[0030] FIG. 11 is a drawing, schematically illustrating a writing operation to the flash memory with a type of big size block.

[0031] FIG. 12 is a drawing, schematically illustrating the block

structure and the mapping table in a flash memory with a type of big size block, according to the preferred embodiment of the invention.

[0032] FIGs. 13-14 are drawings, schematically illustrating a writing operation to the flash memory with a type of big size block, according to the preferred embodiment of the invention.

[0033] FIG. 15 illustrates the comparison between the conventional writing operation and the writing operation of the invention.

[0034] FIG. 16 is a drawing, schematically illustrating a sector structure for a page cache block, according to the preferred embodiment of the invention.

DETAILED DESCRIPTION

[0035] Recently, the big size nonvolatile memory, such as a big size flash memory, has been proposed. For the big size nonvolatile memory, one block has multiple pages and each of the pages has multiple sectors. For example, the page size has 4 physical sectors. In this manner, page is the basic unit for flash programming. From system point of view, the corresponding 4 logical sectors have to be programmed into flash memory at the same time. However, the host side doesn"t always request to write se-

quential 4 logical sector. Eventually, some sequential write in host side may result in a random write so that the whole system performance will be down. This invention proposes a page cache block, for storing the last one page to be written. In this manner, since the page cache block separately stores the page, the frequency of swap operation can be effectively reduced. As a result, the system performance can be effectively improved. An example is provided for descriptions about the features of the invention.

[0036] FIG. 12 is a drawing, schematically illustrating the block structure and the mapping table in a flash memory with a type of big size block, according to the preferred embodiment of the invention. The block structure of the big size nonvolatile memory, according to the invention, includes multiple blocks, like a data block 602, a writing (W) block 604, and a page cache block 610. Also, they correspond to a specific logical block 600. The logical block 600, the data block 602, the writing block 604 are used like the conventional arrangement for the access operation, such as the writing operation. The present invention particularly introduces the page cache block 610 that is associating with the writing block 604, for storing the last page of

data, which is intended to be written to the writing block in the conventional access manner.

[0037] For example, when the sector LBA 0 is to be programmed, the sector LBA 0 accompanying with sectors LBA1 LBA3 as a page 0 is to be written into the writing block 604. However, since this page is the only one page to be written, the page itself is also the last page to be written to the writing block 604. Then, according to the present invention, this page including the sectors LBA0 LBA3 is directly written into the page cache block. Assuming that the writing block 604 and the page cache block are empty at the beginning state, then the page including the data, relating to the sector LBA0 is written into the space of page 0. Then, the mapping table 612 marks the empty page pointer (empty Ptr1) in row 614 to be 1. The use of mapping table has been known by the skilled artisans, and is not further described.

[0038] In next write operation as shown in FIG. 13, when the host requests to write data into any one of the sectors LBA0 LBA3, such as LBA1, since the big size block uses the page as the unit, the page is again written to the page 1 of the page cache block 610. In this situation, since the data stored in sectors LBA0, LBA2, and LBA3 are not

changed, those data are just copied without change. However, in the invention, the swap operation is not necessary but the swap operation is necessary in the conventional method. This kind of situation for the host to sequentially write the sector occurs quite often. Therefore, the invention can effectively reduce the swap operation.

[0039] In general, if the data needs not to cross a page, then the page is directly written into the page cache block. This is because the page by itself is the last page, according to the present invention. For example, if the sectors LBA1 LBA3 are to be written or programmed, the page including the sectors LBA0 LBA3 is directly written into page cache block 610.

[0040] In comparing with the conventional writing operation as shown in FIG. 10 and FIG. 11, the conventional write operation needs a swap between the data block and the writing block, in which the block address No. 5 and No. 200h has been swapped. In the invention, the swap is not necessary.

[0041] Furthermore, in FIG. 14, if another sector LBA9 is requested by the host, then the page 0 and page 1 respectively including the sectors LBA0 LBA3 and LBA4 LBA7 are written to the writing block 604 by copying from D block

602. However, the last page including the sectors LBA8 LBA11 is written to the page cache block at page 2.

[0042] For another situation, for example, the host requests to write 10 sectors (SC=10, SC means the sector count) staring from sector LBA0, after the writing operation in FIG. 13. This situation is usually called a random write, and it needs a swap operation in the conventional method because the overwriting to the previous page, such as page 0, occurs. However, in the invention, since the data spread over three pages, the first two pages are written to the writing block 604 and the last page is written to the page cache block 610. The swap operation is not necessary in the invention.

[0043] In general, when the data has crossed the page, then the front part page(s) is written to the writing block 604, and the last page including at least a portion of the data is written to the page cache block 610. In other words, the last page can be the page itself if the data is not necessary to cross the page, or the last page includes the last four sectors.

[0044] In the present invention, most of the conventional access management can still remain. The only need is to arrange the page cache block to store the latest page with respect

to the writing block. In this manner, the present invention can effectively reduce the frequency of the swapping operation, but is not difficult to be implemented into the conventional big size nonvolatile memory. As a result, the performance of the big size nonvolatile memory with the block structure of the present invention can be effectively improved.

[0045] FIG. 15 shows the improvements of the invention with respect to the previous three step of write operations as the example. In the step 1, the sector LBA0 is requested by sector count (SC) =1. Then, the page 0 is written to the writing block in the conventional method. In the invention, the page 0 is written to the page cache block. The advantages of the invention are not significantly seen yet in step 1. However, in step 2, the sector LBA1 is requested with SC=1. In the conventional method, since the sector LBA1 is still belonging to the page 0, page 0 is necessary to be overwritten, and a swap operation is therefore necessary. In the invention, since the new page 0 as the last page is written to the page cache block, the swap operation is not necessary.

[0046] Further in step 3, a random-write access is requested by the host. For example, 10 sectors (SC=10) are requested

starting from the sector LBA0. Since the sector LBA0 belongs to the page 0, the page 0 should be overwritten. In this situation, the swap operation is necessary for the conventional method. However, in the present, the page0 and page 1 are written to the writing block and the last page 2 is written to the page cache block. There is no overwriting situation occurring. The swap operation in the invention is not necessary.

[0047] FIG. 16 is a drawing, schematically illustrating a structure for a page cache block, according to the preferred embodiment of the invention. The page of the page cache block includes, for example, four sectors. The sector structure includes, for example, 512 bytes as the data area and 16 bytes for the extra area. The extra area stores the basic information, such as logical block number, logical page offset, system flag, ECC, ..., and so on.

[0048] It will be apparent to those skilled in the art that various modifications and variations can be made to the structure of the present invention without departing from the scope or spirit of the invention. In view of the foregoing, it is intended that the present invention covers modifications and variations of this invention provided they fall within the scope of the following claims and their equivalents.